# U.S. NAVAL ACADEMY
# COMPUTER SCIENCE DEPARTMENT
# TECHNICAL REPORT


# Algorithm MinWtBasis for simplifying conjunctions of monomial inequalities

Brown, Christopher W.

USNA-CS-TR-2010-01

January 28, 2010

**USNA Computer Science Dept. ◦ 572M Holloway Rd Stop 9F ◦ Annapolis, MD 21403**

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|---|---|---|
| **22 JAN 2010** | | **00-00-2010 to 00-00-2010** |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| **Algorithm MinWtBasis for simplifying conjunctions of monomial inequalities** | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **U.S. Naval Academy,Computer Science Department,572M Holloway Rd Stop 9F,Annapolis,MD,21403** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES

14. ABSTRACT
**This paper defines a new algorithm ?MinWtBasis? which simplifies conjunctions of monomial inequalities. The simplified equivalent formula produced by MinWtBasis minimizes the sum over all inequalities in the conjunction of the number of non-strict variables appearing, and it runs in polynomial time. For strictly non-strict conjunctions of inequalities this shows that the problem of finding a simplest equivalent formula is in P. This contrasts with the general case and the strict inequality case, in which finding the simplest equivalent formula is NP-Hard.**

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT **unclassified** | b. ABSTRACT **unclassified** | c. THIS PAGE **unclassified** | **Same as Report (SAR)** | **10** | |

# Algorithm MinWtBasis for simplifying conjunctions of monomial inequalities

Christopher W. Brown
Computer Science Department, Stop 9F
United States Naval Academy
572M Holloway Road
Annapolis, MD 21402
wcbrown@usna.edu

January 22, 2010

**Abstract**

This paper defines a new algorithm "MinWtBasis" which simplifies conjunctions of monomial inequalities. The simplified equivalent formula produced by MinWtBasis minimizes the sum over all inequalities in the conjunction of the number of non-strict variables appearing, and it runs in polynomial time. For strictly non-strict conjunctions of inequalities, this shows that the problem of finding a simplest equivalent formula is in P. This contrasts with the general case and the strict inequality case, in which finding the simplest equivalent formula is NP-Hard.

## 1 Introduction

This report builds on the results presented in [1]. That paper gave algorithms for several problems related to computing with conjunctions of monomial inequalities, and proved that the general simplification problem for monomial inequalities is NP-Hard. We will assume the same notation, and will refer to results from that paper frequently, especially Theorem 5.

## 2 MinWtBasis

Suppose $F = A_1 \wedge \cdots \wedge A_m$ is a conjunction of monomial inequalities. Let $B = \{M(A_1), \ldots, M(A_m)\}$. Let "the support of vector $w$", $S(w)$, be the set of indices from the non-strict part at which $w$ is non-zero. Let "the weight of vector $w$", $wt(w)$, be the number of non-zero entries in the non-strict part of $w$, i.e. $|S(w)|$.

**Algorithm 1** MinWtBasis

**Input:** $B$, the set of vectors that are images of the inequalities in formula $F$
**Output:** $B_f$, a minimum-weight set of vectors subject to the constraint that $\bigwedge_{b \in B_f} M^{-1}(N(b))$ is equivalent to $F$

1: $B_f := \{\ \}$
2: $w :=$ a maximum weight element of $B$, if $B = \{\ \}$ or $wt(w) = 0$ return $B_f \cup B$
3: $B := B - \{w\}$
4: $B_{\leq} = \{b \in B \mid S(b) \subseteq S(w)\}$
5: $B_{<} = \{b \in B \mid S(b) \subset S(w)\}$
6: check whether there is a subset $T \subseteq B_{<}$ such that

$$\sum_{t \in T} t \equiv [0, \ldots, 0, 1] \oplus [0, \ldots, 0] \bmod 2$$

if yes, then goto step 2
7: form matrix $M$ over $GF(2)$ whose rows are the elements of $B_{\leq}$ modulo 2
8: do Gaussian elimination on $M$ to put in reduced row echelon form
9: $w' :=$ the result of reducing $w$ mod 2 by the rows of $M$
10: **if** $w'$ or some row of $M$ equals $[0, \ldots, 0, 1] \oplus [0, \ldots, 0]$ **then**
11: add $N(2w + [0, \ldots, 0, 1] \oplus [0, \ldots, 0])$ to $B_f$ (we're adding an equation here!)
12: remove from $B$ any element with support the same as $w$
13: **else if** $w' \neq [0, \ldots, 0] \oplus [0, \ldots, 0]$ **then**
14: $B_f := B_f \cup \{w\}$
15: **end if**
16: goto step 2

## 3 Proof of correctness for MinWtBasis

In this section we prove that the Algorithm MinWtBasis meets its specification, i.e. that it produces a minimum weight set of vectors representing a formula that is equivalent to its input. This requires several lemmas.

Theorem 5 of [1] gives three rules phrased in terms of combining monomial inequalities to produce new monomial inequalities. We note that the rules

trivial translate to equivalent statements about combining vectors (representing inequalities) to produce new vectors (representing inequalities). When we refer to "the rules from Theorem 5", context will make it clear whether the rules as stated or their vector equivalents are intended. It will also be convenient to make the following definition:

**Definition 1** *If $B$ is a set of vectors, then close($B$) is the set of vectors derivable from $B$ using the rules from Theorem 5.*

A few obvious facts about the normalization functions $N$, $\nu$ and $\nu'$:

1. If $w_1 = u_1 \oplus v_1$ and $w_2 = u_2 \oplus v_2$, then $N(w_1+w_2) = \nu(u_1+u_2) \oplus \nu'(v_1+v_2)$.
2. $N(w) = N(N(w))$
3. $N(w_1 + w_2) = N(N(w_1) + N(w_2))$
4. For any vector $u \oplus v$, $u \oplus v + u \oplus v \equiv 0 \oplus 0 \pmod 2$

**Lemma 1** *If vector $w$ is derivable from $B$ using rules 1 and 3 from Theorem 5, then for some $S \subseteq B$ and vector $v'$,*

$$N(w) = N(\sum_{w' \in S} w' \ + 2v').$$

Proof. We proceed inductively on the number of steps in the derivation. Clearly the lemma holds for 0 steps, with $w' = w$ and $v' = 0$. Consider a derivation of $k + 1$ steps.

Case 1: the last step is an application of rule 3, i.e. $w = u \oplus v \ + \ 2v''$, for some $v''$, where $u \oplus v$ is derivable from $B$ in $k$ steps. Thus, by induction, for some $S' \subseteq B$ and vector $v'$

$$N(u \oplus v) = N(\sum_{w' \in S} w' \ + 2v').$$

Thus,

$$N(w) = N(u \oplus v \ + \ 2v'') = N(\sum_{w' \in S} w' \ + 2v' + 2v'') = N(\sum_{w' \in S} w' \ + 2(v' + v''))$$

and we are done.

Case 2: the last step is an application of rule 1, i.e. $w = w_1 + w_2$ where $w_1$ and $w_2$ are each derivable in $k$ or fewer steps. Thus, by induction, $w_1 =$

$N(\sum_{w' \in S_1} w' + 2v'_1)$ and $w_2 = N(\sum_{w' \in S_2} w' + 2v'_2)$. So

$$\begin{array}{rcl} N(w) & = & N(w_1 + w_2) = N(\sum_{w' \in S_1} w' + 2v'_1 + \sum_{w' \in S_2} w' + 2v'_2) \\ & = & N(\sum_{w' \in (S_1 \cup S_2) - (S_1 \cap S_2)} w' + 2\sum_{w' \in (S_1 \cap S_2)} w' + 2v'_1 + 2v''_2) \\ & = & N(\sum_{w' \in (S_1 \cup S_2) - (S_1 \cap S_2)} w' + 2(\sum_{w' \in (S_1 \cap S_2)} w' + v'_1 + v''_2)) \end{array}$$

□

**Lemma 2** *If $S(q) = S(w)$ and, for some $p$, $N(p+q) = N(w)$, then $N(p+w) = N(q)$.*

Proof. Obvious. □

**Theorem 1** *Let $F$ be as in Theorem 5 — satisfiable, $F = A_1 \wedge \cdots \wedge A_m$. Let $A$ be an atomic formula $\prod_{j=1}^{n} x_j^{d_j}\ \sigma\ 0$. Let $M(A) = u \oplus v$, then $F \Rightarrow A$ if and only if there is a subset $U = \{u_1 \oplus v_1, \ldots, u_r \oplus v_r\} \subseteq \{M(A_1), M(A_2), \ldots, M(A_m)\}$, such that $S(u_i \oplus v_i) \subseteq S(M(A))$ for each each $u_i \oplus v_i \in U$, and either the elements of $U$ sum to $[0, \ldots, 0, 1] \oplus [0, \ldots, 0]$ modulo 2 or*

$$\sum_{i=1}^{r} u_i \oplus v_i \equiv u \oplus v \pmod 2.$$

Proof. The backwards direction of this theorem is obvious. It follows directly from Theorem 5. So we consider the forward direction. Note that $F \Rightarrow A$ if and only if there is a derivation of $M(A)$ from $\{M(A_1), M(A_2), \ldots, M(A_m)\}$ using the rules of Theorem 5. Each rule of Theorem 5 takes two vectors and combines them, producing a new vector whose support is the union of the supports of the original two vectors. Thus, no vector whose support includes an element not in the support of $M(A)$ can be involved in the derivation. This justifies the requirement that $S(u_i \oplus v_i) \subseteq S(M(A))$ for each each $u_i \oplus v_i \in U$. Next we note that if there is a derivation that uses only rules 1 and 3, then Lemma 1 clearly implies this theorem — in fact it implies the second case of this theorem's conclusion. Therefore, suppose that $M(A)$ is such that any derivation requires an application of rule 2. Consider the first application of rule 2 in such a derivation. The rule requires a vector $[0, \ldots, 0, 1] \oplus v$ that is implied by $\{M(A_1), M(A_2), \ldots, M(A_m)\}$, where $v \neq 0$ but $v \equiv 0 \pmod 2$. By our assumption, $v$ must be derivable using only rules 1 and 3. By Lemma 1, there is a subset of $U$ whose sum is equivalent to $[0, \ldots, 0, 1] \oplus v$ modulo 2, and thus is equivalent to $[0, \ldots, 0, 1] \oplus [0, \ldots, 0]$ modulo 2. □

**Lemma 3** *If for some subset* $U = \{u_1 \oplus v_1, \ldots, u_r \oplus v_r\} \subseteq \{M(A_1), M(A_2), \ldots, M(A_m)\}$

$$\sum_{i=1}^{r} u_i \oplus v_i \equiv (u \oplus v) + [0, \ldots, 0, 1] \oplus [0, \ldots, 0] \pmod 2$$

*then*

$$F \wedge A \Leftrightarrow F \wedge \prod_{x_i \in S(U)} x_i = 0,$$

*where* $S(U) = \cup_{b \in U} S(b)$.

PROOF. Obvious given Theorem 5. □

**Theorem 2** *Algorithm MinWtBasis terminates with output* $B_f$ *meeting its specifications: i.e.* $B_f$ *is a minimum-weight set of vectors subject to the constraint that* $\bigwedge_{b \in B_f} M^{-1}(N(b))$ *is equivalent to* $F$.

PROOF. To prove the correctness of a greedy algorithm, i.e. that it produces an optimum solution, it suffices to prove (1) that its greedy choice is always part of some optimum solution, and (2) that the problem has the optimum subproblem property (see Chapter 16 of [2] for a discussion of correctness proofs for greedy algorithms).

This algorithm is essentially a big loop from Step 2 to Step 16. Each time through the loop we choose an element maximum weight element $w$ from $B$, remove it, and then make one of a number of choices. We distinguish each choice as a separate case, and prove (1) and (2) for each case separately.

**Case 1:** The condition at Step 6 is met. In this case we do not add $w$ to $B_f$, we simply jump to the top of the loop with $B$ now diminished by having removed $w$. Suppose $B'_f$ is an optimum solution to the original problem.

1. $B'_f$ does not contain $w$. Suppose it did. For each $t \in T$ some subset of $B'_f$ sums to $t$. Moreover, none of these subsets contain $w$ since the support of each $t$ is a strict subset of the support of $w$. The sum of the sums of these subsets of $B'_f$ is an equation with support contained in $S(w)$, so $w$ is derivable from $B'_f - \{w\}$, contradicting the optimality of $B'_f$. Thus, $B'_f$ does not contain $w$.
2. Clearly, $B'_f$ is an optimum solution to the the subproblem $B - \{w\}$ as well as an optimum solution to the original problem $B$.

**Case 2:** The "then" clause of the "if" at Step 10. In this case, we add equation $N(2w + [0, \ldots, 0, 1] \oplus [0, \ldots, 0])$ to $B_f$ and remove from $B$ any element with support the same as $w$.

1. Suppose $B'_f$ is an optimum solution that does not contain $N(2w+[0,\ldots,0,1]\oplus [0,\ldots,0])$. Claim: $B'_f$ must contain an element $z$ such that $S(z) = S(w)$. Let $T' \subseteq B'_f$ be such that $N(2w+[0,\ldots,0,1]\oplus[0,\ldots,0]) \in close(B_<)$. $T'$ contains an element not in $close(B_<)$, since otherwise we would be in Case 1. Thus, such an element is generated from $B$ using some vector from $B$ with support equal to $S(w)$, which means that the element has support that contains $S(w)$ and, since the element is used to derive a vector with support equal to $S(w)$, we conclude that the element's support is exactly $S(w)$. This proves the claim.

   Let $z$ then be an element of $B'_f$ such that $S(z) = S(w)$. Since $N(2w + [0,\ldots,0,1] \oplus [0,\ldots,0])$ generates all vectors with support equal to $S(w)$,

$$B''_f = B'_f - \{z\} \cup \{N(2w + [0,\ldots,0,1] \oplus [0,\ldots,0])\}$$

   is an optimum solution and, moreover, is an optimum solution that contains the "greedy choice" from this case.

2. Clearly, an optimum solution to

$$B - (B_{\leq} - B_<) \cup \{N(2w + [0,\ldots,0,1] \oplus [0,\ldots,0])\}$$

   is an optimum solution to $B$, and for any optimum solution $B''_f$ containing $N(2w+[0,\ldots,0,1]\oplus[0,\ldots,0])$, $B''_f - \{N(2w+[0,\ldots,0,1]\oplus[0,\ldots,0])\}$ is an optimum solution to $B - (B_{\leq} - B_<)$, which is what we continue with after our greedy choice.

**Case 3:** The "then" clause of the "else if" on line 13. Since we are not in Case 1 or Case 2, no equation with support contained in $S(w)$ can be derived from the elements of $B$ (including $w$). Furthermore, since we are in this case, no subset of $B - \{w\}$ sums to $w$ modulo 2, which by Theorem 1 means $B - \{w\}$ does not generate $w$.

In this case, $w$ is added to $B_f$.

1. First we must prove that some optimum solution contains $w$. Suppose $B'_f$ is an optimum solution that does not contain $w$. Since $B'_f$ generates $w$ but $B-\{w\}$ does not, there must be some $p \in B'_f$ such that $p \notin close(B-\{w\})$. However, $p \in close(B)$, so since $B$ implies no equations with support contained in $S(w)$, $p$ must be derivable from $B$ using only rules 1 and 3 of Theorem 5. Thus, by Lemma 1 for some $T \subseteq B - \{w\}$ and some vector $v'$

$$N(p) = N(w + \sum_{w' \in T} w' \ + \ 2v').$$

   Thus $S(w) \subseteq S(p)$. If $S(w) \subset S(p)$, each element of $T$ is generated by $B'_f - \{p\}$ — since $w$ is a maximum weight element of $B$ — and so $B'_f -$

$\{p\} \cup \{w\}$ is an optimum solution. Otherwise $S(w) = S(p)$, which means each element of $T$ has support contained in $S(w)$. Since each element of $T$ is expressible as a sum of elements of $B'_f$ (possibly plus some vector of the form $2v'$), we can write $N(p)$ as

$$N(p) = N(w + \sum_{w' \in T''} w' \ + \ 2v'').$$

for some $T' \subseteq B'_f$ and some vector $v''$. We distinguish two cases: a) if $p \notin T''$, in which case $B'_f - \{p\} \cup \{w\}$ is clearly an optimum solution. b) if $p \in T''$ then

$$N(p) = N(w + p + \sum_{w' \in T'' - \{p\}} w' \ + \ 2v'')$$

and by Lemma 2

$$N(w) = N(p + p + \sum_{w' \in T'' - \{p\}} w' \ + \ 2v'') = N(\sum_{w' \in T'' - \{p\}} w' \ + \ 2v''')$$

Thus, for some $q \in T'' - \{p\}$ we have $S(q) = S(w)$, since otherwise each element of $T'' - \{p\}$ would be generated by $B - \{w\}$, implying that $w$ is generated by $B - \{w\}$, which is as a contradiction. By Lemma 2,

$$N(q) = N(w + \sum_{w' \in T'' - \{p,q\}} w' \ + \ 2v''')$$

and so clearly $B'_f - \{q\} \cup \{w\}$ is an optimum solution.

2. Next we must prove the optimum subproblem property.

   Claim 1: There is an optimum solution $B'_f$ such that $w \in B'_f$ and $B'_f - \{w\} \subseteq close(B - \{w\})$. Let $p \in B'_f - \{w\}$. By optimality, if $p$ is an equation, it must be a minimal equation. Clearly, $p \in close(B)$. Suppose $p \notin close(B - \{w\})$. By minimality, $p$ must be derivable using only rules 1 and 3 of Theorem 5. Thus, by Lemma 1, for some $T \subseteq B - \{w\}$

$$N(p) = N(w + \sum_{t \in T} t \ + \ 2v')$$

   So $S(w) \subseteq S(p)$. In fact $S(w) \subset S(p)$ is not possible, because $w$ has maximum weight in $B$, so $B'_f - \{p\}$ generates all of $B$, making $p$ extraneous, and contradicting the optimality of $B'_f$. Thus $S(p) = S(w)$, so $N(N(p + w) + w) = N(p)$. Thus, we may replace $p$ in $B'_f$ with $N(p + w)$ and the closure remains the same and so does the weight, but now $p \in close(B - \{w\})$. If all such $p$ are replaced by $p + w$, we get an optimum solution meeting the requirement.

Claim 2: Assuming $B'_f$ is an optimal solution satisfying Claim 1, $B-\{w\} \subseteq close(B'_f - \{w\})$. Suppose not. Then for some $p \in B - \{w\}$

$$N(p) = N(\sum_{t \in T \subseteq B'_f - \{w\}} t \ + \ 2v' \ + \ w).$$

Therefore, $S(w) \subseteq S(p)$. But $w$ has maximum weight in $B$, so $S(p) = S(w)$. By Lemma 2

$$N(w) = N(\sum_{t \in T \subseteq B'_f - \{w\}} t \ + \ 2v' \ + \ p).$$

and by Claim 1, each $t$ is generated by $B-\{w\}$, so $w$ is actually generated by $B-\{w\}$ which contradicts the assumption that we are in Case 3 of the algorithm.

Thus there is an optimum solution $B'_f$ such that $close(B-\{w\}) = close(B'_f - \{w\})$, so $B'_f - \{w\}$ is an optimum solution to problem $B - \{w\}$.

**Case 4:** None of the "if"s apply. In this case, the greedy choice is simply to remove $w$ from $B$ because $w'$, the result of reducing $w$ by the rows of $M$ is zero modulo 2. By Theorem 1, $w$ is derivable from $B - \{w\}$. Suppose $B'_f$ is an optimum solution that contains $w$. No subset of $B - \{w\}$ generates an equation since, otherwise, we would be in case 1 or case 2. Thus, $w$ is generated from $B - \{w\}$ using only rules 1 and 3 of Theorem 5, so

$$N(w) = N(\sum_{w \in S} w' \ 2v'), \text{ where } S \subseteq B - \{w\}.$$

If no element of $S$ has support equal to $S(w)$, then each element of $S$ is generated by $B'_f - \{w\}$, so $B'_f - \{w\}$ generates $w$, contradicting the optimality of $B'_f$. Otherwise, let $Z = \{z \in S | S(z) = S(w)\}$, so

$$N(w) = N(x + (\sum_{w' \in S - Z} w' \ + \ 2v')), \text{ where } x = \sum_{z \in Z} z.$$

Each element of $S - Z$ is generated by $B'_f - \{w\}$, so $w \in close(B'_f - \{w\} \cup \{x\})$. Since $x \in close(B'_f)$, we have $close(B'_f) = close(B'_f - \{w\} \cup \{x\})$, and because $w$ and $x$ have the same weight, an optimum solution for $B - \{w\}$ is an optimum solution for $B$. □

# 4 Conclusion

This report has introduced the new algorithm, MinWtBasis, which simplifies monomial inequalities so that the non-strict part is minimal. The algorithm

clearly runs in polynomial time, which shows that strictly non-strict conjunctions of monomial inequalities can be found in polynomial time, in contrast to the general problem of simplification of conjunctions of monomial inequalities, which is NP-Hard.

# References


[1] Brown, C. W. Fast simplifications for tarski formulas. In *ISSAC '09: Proceedings of the 2009 international symposium on Symbolic and algebraic computation* (New York, NY, USA, 2009), ACM, pp. 63–70.

[2] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms, 2nd edition*. MIT Press, McGraw-Hill Book Company, 2000.